UNIVERSITY OF SASKATCHEWAN

Department of Computer Science
176 Thorvaldson Building
110 Science Place, Saskatoon, SK, S7N 5C9, Canada
Telephine: (306) 966-4886, Facimile: (306) 966-4884

Assignment 1

Java Basics, Object Types and Features

Date Due: May 15, 2019, 8pm **Total Marks: 65**

General Instructions

- **This assignment is individual work.** You may discuss questions and problems with anyone, but the work you hand in for this assignment must be your own work.
- **Assignments are being checked for plagiarism.** We are using state-of-the-art software to compare every pair of student submissions.
- **Make sure your name and student number appear at the top of every document you hand in.** These conventions assist the markers in their work. Failure to follow these conventions will result in needless effort by the markers, and a deduction of grades for you.
- **Assignments must be submitted to Moodle.**
- **Moodle will not let you submit work after the assignment deadline.** It is advisable to hand in each answer that you are happy with as you go. You can always revise and resubmit as many times as you like before the deadline; only your most recent submission will be graded.
- **Programming questions must be written in Java.**
- **Non-programming questions must be submitted as either `.txt` or `.pdf` files.** If other file types are used, you will receive a grade of zero for that question.

Version History

- **07/05/2019**: released to students

Question 1 (20 points):

Purpose: Practising Java Syntax and Programming

Degree of Difficulty: Easy.

(a) (5 points) Implement a Java class called `ArrayAverager` that uses a method named `average` to return the average value of an array of `type` double, as done in this python snippet:

```
def Average(S):
     total = 0
     for x in S:
          total = total + X
     average = total/len(S)
     return average
```

On moodle you will find a starter file titled `ArrayAverager.java` with the `main` method already implemented. Your task is to complete the TODO's in this file. Once these are correctly implemented your program should display a value of `3.25` to the console.

(b) (5 points) Write a Java class named `Greeter` that implements a method named `introductions`. This method should prompt the user for their name (using the `java.util.Scanner` class), displays a greeting, and return the name as a String, as done in this python snippet:

```
def introductions(greeting):
     print(greeting)
     x = input('Please enter your name: ')
     print('Hello, ' + str(x))
     return x

username = introductions('Welcome')
```

The `main` method of this class should invoke `introductions` and save the result in a class attribute.

(c) (5 points) Write a java class named `CapitalCounter` that implements a method named `countCaps` to return the number of capital letters within a String, as done in this python snippet
(hint: use `Character.isUpperCase()`):

```
def countCaps(s):
     count = 0
     for character in s:
          if character.isupper():
               count = count + 1
     return count
```

Add three demonstrations of this function to the `main` method of the `CapitalCounter` class. The results should be printed to the console. Choose input strings that show your function works with different types of String input in terms of equivalence classes.

(d) (5 points) Write a class named `NumberGuesser` that implements the following code within its main method by using a loop and the `java.util.Scanner` class:

```
guess = int(input('Guess a nuber between 1 and 100: '))
while guess < 1 or guess > 100:
     if guess < 1:
          print('Too low!')
     else:
          print('Too high!')
```

```
    guess = int(input('Guess a nuber between 1 and 100: '))

print('Valid guess!')
```

What to Hand In

- The completed `ArrayAverager.java` program.
- The completed `Greeter.java` program.
- The completed `CapitalCounter.java` program.
- The completed `NumberGuesser.java` program.

Be sure to include your name, NSID, student number and course number at the top of all documents.

Evaluation

For full marks, your java programs must compile and run without error to produce the desired output. 1 mark will be deducted per compiler and/or run-time error.

Question 2 (45 points):

Purpose: Practising Object-Oriented Thinking.

Degree of Difficulty: Moderate.

The objective of this question is to get familiar with objects, and modelling with object types (classes). You will not write any code for this question, rather you will analyze three applications to determine what object types and containers are needed, what fields are needed for each object, and what item access is needed for each container.

For each application provide:

- The object types/classes involved, with the main fields/attributes for each class.
- The containers needed, with the type of the items stored in the container, and how the items are accessed from the container, e.g., in order first to last, by index, by field value, by insertion order, etc. You do not need to include how the items are stored (array, linked list, stack, tree, etc).
- Any important methods, including getters and setters when necessary. Make sure method names are descriptive.

For this question, since you are not writing any code, submitted files must be either `.txt` or `.pdf` files. **If other file types are used, you will receive a grade of zero for this question.**

In your file, list the name of a class, followed by its fields (and then its methods). This is not programming, so you do not need to use precise syntax. Note that it is often the case that an object of type A contains another object of some type B. In this situation, A will be shown as having a field of type B, and type B will be listed and described separately. Do not mix the descriptions for A and for B. If the purpose of a type or field is not obvious from its name, include a brief English description of the type or field.

Don't worry about how input or output is handled, or the overall control of the application. Just consider what objects and capabilities are needed for each class.

(a) (15 points) **Application 1: Flight Reservation System**
Consider a flight reservation system that needs to keep track of passengers and their flights. A passenger can have several flights booked at a time, and a flight will have several passengers. The booking of a flight by a passenger can include seat selection. It is also important to keep track of a flights arrival and departure times. As well as a method to identify if there are any time conflicts in an individual passenger's flight plan (i.e. if an earlier flight arrives after the later flight departs).

(b) (15 points) **Application 2: A Doggie Daycare**
Doggie Daycare's are businesses that allow owners to drop off their dogs for a day of fun while they have to work. Doggy Daycares have a maximum number of dogs that they can care for in a single day. Dogs can be dropped off for either a half-day (morning/afternoon) or a full day. It is important to keep track of when the dog will be picked up, who their owner(s) are and any special information such as dietary restrictions. Additionally, the facility needs to know which staff members are working on a given day. It should also have a way to keep track of owners, their contact information and a way to indicate which dog(s) belong to them.

(c) (15 points) **Application 3: An Emergency Room**
Consider an emergency room ward in a hospital with a fixed number of beds. The objective is to keep track of which patient is in each bed of the ward, and the doctor or doctors for the patient. Also, it should be easy to determine which beds are empty so that a new patient can be given an empty bed. Finally, a doctor should know which patients are theirs. Methods should exists that allow patients to be admitted and discharged from the emergency ward.

What to Hand In

- A text file named `a1q2.txt` or `a1q2.pdf`. Your answers to each of the above applications must be clearly distinguishable in the file.

Reminder: Submitted files must be either `.txt` or `.pdf` files. **If other file types are used, you will receive a grade of zero for this question.**

Be sure to include your name, NSID, student number and course number at the top of all documents.

Evaluation

Each application will be graded according to the following scheme:

- 5 marks: For correctly identifying what Objects and attributes are necessary to represent the application.
- 5 marks: For correctly identifying the required container(s) in each class. As well as describing what type of data will be stored and how it will be accessed.
- 5 marks: For the class methods.